UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
09/608,549	06/30/2000	Gurumukh S. Tiwana	CISCP151	2362

22434 7590 11/30/2005

BEYER WEAVER & THOMAS LLP
P.O. BOX 70250
OAKLAND, CA 94612-0250

EXAMINER	
DUONG, THOMAS	
ART UNIT	**PAPER NUMBER**
2145	

DATE MAILED: 11/30/2005

Please find below and/or attached an Office communication concerning this application or proceeding.

Office Action Summary	Application No.	Applicant(s)	
	09/608,549	TIWANA ET AL.	
	Examiner	Art Unit	
	Thomas Duong	2145	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) OR THIRTY (30) DAYS, WHICHEVER IS LONGER, FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on *02 September 2005*.

2a)☐ This action is **FINAL**. 2b)☒ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) *1-2, 4-8, 11-20, 22, 24-26, 28-32, and 34-41* is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) *1-2, 4-8, 11-20, 22, 24-26, 28-32, and 34-41* is/are rejected.

7)☐ Claim(s) _____ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☐ The specification is objected to by the Examiner.

10)☐ The drawing(s) filed on _____ is/are: a)☐ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1) ☐ Notice of References Cited (PTO-892)
2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)
3) ☐ Information Disclosure Statement(s) (PTO-1449 or PTO/SB/08) Paper No(s)/Mail Date _____.
4) ☐ Interview Summary (PTO-413) Paper No(s)/Mail Date. _____.
5) ☐ Notice of Informal Patent Application (PTO-152)
6) ☐ Other: _____.

DETAILED ACTION

Request for Continued Examination

1. A request for continued examination under 37 CFR 1.114, including the fee set forth in 37 CFR 1.17(e), was filed in this application after final rejection. Since this application is eligible for continued examination under 37 CFR 1.114, and the fee set forth in 37 CFR 1.17(e) has been timely paid, the finality of the previous Office action has been withdrawn pursuant to 37 CFR 1.114.

2. Amendment received September 2, 2005 has been entered into record. *Claims 1-2, 4-8, 11-20, 22, 24-26, 28-32, and 34-41* remain pending.

Response to Amendment

3. This office action is in response to the applicants Amendment filed on September 2, 2005. Applicant amended *claims 1, 5-8, 17-18, 20, 22, 24-25, 29-32, and 40-41* and canceled *claims 3, 10, 21, 23, 27, and 33. Claims 1-2, 4-8, 11-20, 22, 24-26, 28-32, and 34-41* are presented for further consideration and examination.

Claim Rejections - 35 USC § 103

4. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

5. *Claims 1-2, 4-8, 11-20, 22, 24-26, 28-32, and 34-41* are rejected under 35 U.S.C. 103(a) as being unpatentable over Li (US005634125) and in view of Dreszer (US006442661B1).

6. With regard to *claims 1, 18, 25 and 41*, Li discloses,

- *a) when a new cache system starts up in a cache cluster having a plurality of total buckets, determining a full bucket allocation for the new cache system;* (Li, abstract; col.1, lines 18-37, lines 38-54; col.2, lines 29-44, lines 48-61; col.6, lines 18-51; col.6, line 52 - col.7, line 53; col.9, lines 4-21; fig.6A; fig.9)
  Li teaches of a method for *"data redistribution process for adding a new node"* (Li, col.6, lines 18-19) to a database system *"where the new node is physically attached and registered to the parallel database network"* (Li, col.6, lines 20-21) and *"the buckets of data to be moved to the new node are determined for each existing node"* (Li, col.6, lines 24-25). Hence, when a new node is added to the system, a data redistribution process takes place, wherein a portion of the buckets of data for each existing node redistributes to the new node.
- *b) periodically determining a load of the new cache system;* (Li, col.1, lines 18-37-54; col.2, lines 29-44, lines 48-61; col.6, lines 18-51; col.6, line 52 - col.7, line 53; col.9, lines 4-21; fig.6A; fig.9)
  Li teaches that *"in the quiescent-mode operation, load balancing is the primary goal"* (Li, col.6, lines 52-53) implying that is very important to maintain a balanced workload among the existing nodes. Also, according to Li, *"while the description above has concentrated on a redistributing data when a new node is added into the parallel database system, the invention may also be used when the PDB*

system becomes imbalanced" (Li, col.9, lines 4-7). Hence, Li not only teaches of redistributing the data buckets when a new node is added, but also using the technique once the system becomes imbalanced and thus requires a redistribution of the data buckets. Furthermore, Li admits as prior art *"eventually, the system will become imbalanced across the nodes. Thus, the data will occasionally have to be redistributed to rebalance the load"* (Li, col.1, lines 50-52).

- *c) each time it is periodically determined that the new cache system is underloaded and buckets have not been previously shed from the new cache system,* (Li, col.1, lines 18-37-54; col.2, lines 29-44, lines 48-61; col.6, lines 18-51; col.6, line 52 - col.7, line 53; col.9, lines 4-21; fig.6A; fig.9)
  Li teaches that *"in the quiescent-mode operation, load balancing is the primary goal"* (Li, col.6, lines 52-53) implying that is very important to maintain a balanced workload among the existing nodes. Also, according to Li, *"while the description above has concentrated on a redistributing data when a new node is added into the parallel database system, the invention may also be used when the PDB system becomes imbalanced"* (Li, col.9, lines 4-7). Hence, Li not only teaches of redistributing the data buckets when a new node is added, but also using the technique once the system becomes imbalanced and thus requires a redistribution of the data buckets. Furthermore, Li admits as prior art that *"eventually, the system will become imbalanced across the nodes. Thus, the data will occasionally have to be redistributed to rebalance the load"* (Li, col.1, lines 50-52).

- *d) each time it is periodically determined that the new cache system is underloaded and buckets have been shed previously from the new cache system,* (Li, col.1, lines 18-37-54; col.2, lines 29-44, lines 48-61; col.6, lines 18-51; col.6, line 52 - col.7, line 53; col.9, lines 4-21; fig.6A; fig.9)
  Li teaches that *"in the quiescent-mode operation, load balancing is the primary goal"* (Li, col.6, lines 52-53) implying that is very important to maintain a balanced workload among the existing nodes. Also, according to Li, *"while the description above has concentrated on a redistributing data when a new node is added into the parallel database system, the invention may also be used when the PDB system becomes imbalanced"* (Li, col.9, lines 4-7). Hence, Li not only teaches of redistributing the data buckets when a new node is added, but also using the technique once the system becomes imbalanced and thus requires a redistribution of the data buckets. Furthermore, Li admits as prior art that *"eventually, the system will become imbalanced across the nodes. Thus, the data will occasionally have to be redistributed to rebalance the load"* (Li, col.1, lines 50-52).
- *e) each time it is periodically determined that the new cache system is overloaded,* (Li, col.1, lines 18-37-54; col.2, lines 29-44, lines 48-61; col.6, lines 18-51; col.6, line 52 - col.7, line 53; col.9, lines 4-21; fig.6A; fig.9)
  Li teaches that *"in the quiescent-mode operation, load balancing is the primary goal"* (Li, col.6, lines 52-53) implying that is very important to maintain a balanced workload among the existing nodes. Also, according to Li, *"while the description above has concentrated on a redistributing data when a new node is added into the parallel database system, the invention may also be used when the PDB*

system becomes imbalanced" (Li, col.9, lines 4-7). Hence, Li not only teaches of redistributing the data buckets when a new node is added, but also using the technique once the system becomes imbalanced and thus requires a redistribution of the data buckets. Furthermore, Li admits as prior art that *"eventually, the system will become imbalanced across the nodes. Thus, the data will occasionally have to be redistributed to rebalance the load"* (Li, col.1, lines 50-52).

- *wherein each bucket portion corresponds to a portion of the total traffic being handled by the cache cluster.* (Li, abstract; col.1, lines 18-37, lines 38-54; col.2, lines 29-44, lines 48-61; col.6, lines 18-51; col.6, line 52 - col.7, line 53; col.9, lines 4-21; fig.6A; fig.9)

  Li teaches of a method for *"data redistribution process for adding a new node"* (Li, col.6, lines 18-19) to a database system *"where the new node is physically attached and registered to the parallel database network"* (Li, col.6, lines 20-21) and *"the buckets of data to be moved to the new node are determined for each existing node"* (Li, col.6, lines 24-25). Hence, when a new node is added to the system, a data redistribution process takes place, wherein a portion of the buckets of data for each existing node redistributes to the new node.

However, Li does not teach,

- *a) and assigning a portion of the full bucket allocation to the new cache system that was determined at start up;*
- *c) slowly assigning a portion of the full bucket allocation that was determined at startup unless the cache cluster is operating at a maximum load;*

- *d) slowly re-assigning a portion of the buckets that have been previously shed from the new cache system to the new cache system unless the cache cluster is operating at a maximum load; and*
- *e) shedding a portion of the buckets previously assigned to the new cache system,*

Dreszer teaches,

- *a) and assigning a portion of the full bucket allocation to the new cache system that was determined at start up;* (Dreszer, col.2, line 65 – col.3, line 16; col.3, lines 49-60; col.4, line 49 – col.5, line 8; col.5, lines 42-65; col.5, line 66 – col.6, line 8; col.6, lines 9-31; col.6, line 55 – col.7, line 15; col.8, lines 1-55; fig.10-12; col.7, lines 61-65; col.12, lines 49-67; modules 70-72, fig.4)

  Dreszer teaches a method of *"[allocating] of long term memory from the heap 34 during startup"* (Dreszer, col.8, lines 27-28) for a memory management system and, in particular, for a system cache buffer environment by *"adjusting the size in the found/existing allocation header 98 of the found block 95 (e.g., subtracting the requested size from the size of the allocation header 98) and creating a new allocation header 98a in memory beyond the portion 95a of the block 95 for the remaining free portion 95b of the found block 95"* (Dreszer, col.8, lines 42-47) and *"marking the existing allocation header 98 for said memory portion 95a as 'allocated' (not free)"* (Dreszer, col.8, lines 51-52). Hence, Dreszer teaches of allocating only portions of the available memory as necessary or when requested and adjusting the unallocated or the still available memory accordingly.
- *c) slowly assigning a portion of the full bucket allocation that was determined at startup unless the cache cluster is operating at a maximum load;* (Dreszer, col.2,

line 65 – col.3, line 16; col.3, lines 49-60; col.4, line 49 – col.5, line 8; col.5, lines 42-65; col.5, line 66 – col.6, line 8; col.6, lines 9-31; col.6, line 55 – col.7, line 15; col.8, lines 1-55; fig.10-12; col.7, lines 61-65; col.12, lines 49-67; modules 70-72, fig.4)

Dreszer teaches a method of *"[allocating] of long term memory from the heap 34 during startup"* (Dreszer, col.8, lines 27-28) for a memory management system and, in particular, for a system cache buffer environment by *"adjusting the size in the found/existing allocation header 98 of the found block 95 (e.g., subtracting the requested size from the size of the allocation header 98) and creating a new allocation header 98a in memory beyond the portion 95a of the block 95 for the remaining free portion 95b of the found block 95"* (Dreszer, col.8, lines 42-47) and *"marking the existing allocation header 98 for said memory portion 95a as 'allocated' (not free)"* (Dreszer, col.8, lines 51-52). Hence, Dreszer teaches of allocating only portions of the available memory as necessary or when requested and adjusting the unallocated or the still available memory accordingly.

- *d) slowly re-assigning a portion of the buckets that have been previously shed from the new cache system to the new cache system unless the cache cluster is operating at a maximum load; and* (Dreszer, col.2, line 65 – col.3, line 16; col.3, lines 49-60; col.4, line 49 – col.5, line 8; col.5, lines 42-65; col.5, line 66 – col.6, line 8; col.6, lines 9-31; col.6, line 55 – col.7, line 15; col.8, lines 14-25; fig.10-12; col.7, lines 61-65; col.12, lines 49-67; modules 70-72, fig.4)

  Dreszer teaches of *"increasing/decreasing size queues in relation to memory requests (tuning) and performing trimming of size queues"* (Dreszer, col.7, lines 63-65) for a memory management system and, in particular, for a system cache

buffer environment. Furthermore, Dreszer teaches of *"periodically reorganize/trim the size queue and attempt to release file system cache buffers back to the file system"* (Dreszer, col.12, lines 56-58).

- *e) shedding a portion of the buckets previously assigned to the new cache system,* (Dreszer, col.2, line 65 – col.3, line 16; col.3, lines 49-60; col.4, line 49 – col.5, line 8; col.5, lines 42-65; col.5, line 66 – col.6, line 8; col.6, lines 9-31; col.6, line 55 – col.7, line 15; col.8, lines 14-25; fig.10-12; col.7, lines 61-65; col.12, lines 49-67; modules 70-72, fig.4)

  Dreszer teaches of *"increasing/decreasing size queues in relation to memory requests (tuning) and performing trimming of size queues"* (Dreszer, col.7, lines 63-65) for a memory management system and, in particular, for a system cache buffer environment. Furthermore, Dreszer teaches of *"periodically reorganize/trim the size queue and attempt to release file system cache buffers back to the file system"* (Dreszer, col.12, lines 56-58).

Therefore, it would have been obvious to one of ordinary skill in the art at the time of the invention was made to the teachings of Dreszer with the teachings of Li to provide rapid memory allocation and de-allocation, reduced memory fragmentation, maximizes the amount of memory available for a cache (e.g., file system I/O buffers) while optimizing the amount of memory available for other uses, and manages competition for different memory uses by system self-adaptation to different usage levels across different network environments and over time within one network environment, including self-tuning to optimize performance to a variety of environments and dynamic conditions.

7. With regard to *claims 2, 7-8, 12-14, 19, 24, 26, 31-32, and 35-37*, Li and Dreszer disclose,

- *assigning the full bucket allocation to the new cache system when the cache cluster is operating at a maximum load* (Li, abstract; col.1, lines 18-37, lines 38-54; col.2, lines 29-44, lines 48-61; col.6, lines 18-51; col.6, line 52 - col.7, line 53; col.9, lines 4-21; fig.6A; fig.9)

8. With regard to *claim 20*, Li and Dreszer disclose,

- *wherein slowing assigning a portion of the full bucket allocation to the new cache comprises: initials assigning a portion of the full bucket allocation to the new cache system; when no buckets have been previously shed, assigning a portion of the unassigned buckets to the new cache system; and when buckets have been previously shed, assigning a portion of a number of buckets that were previously shed from the new cache system.* (Dreszer, abstract; col.2, line 65 – col.3, line 16; col.3, lines 49-60; col.4, line 49 – col.5, line 8; col.5, lines 42-65; col.5, line 66 – col.6, line 8; col.6, lines 9-31; col.6, line 55 – col.7, line 15; col.8, lines 14-25; fig.10-12)

9. With regard to *claims 4-6, 11, 22, 28-30, and 34*, Li and Dreszer disclose,

- *wherein shedding a portion of tree buckets previously assigned to the new cache comprises: when no buckets have been previously shed, periodically shedding a portion of the assigned buckets from the new cache system; when buckets have been previously shed, periodically shedding a portion of a number of buckets that were previously shed from the new cache system;* (Dreszer, abstract; col.2, line

65 – col.3, line 16; col.3, lines 49-60; col.4, line 49 – col.5, line 8; col.5, lines 42-65; col.5, line 66 – col.6, line 8; col.6, lines 9-31; col.6, line 55 – col.7, line 15; col.8, lines 14-25; fig.10-12; col.7, lines 61-65; col.12, lines 49-67; modules 70-72, fig.4)

10. With regard to *claims 15-17, and 38-40*, Li and Dreszer disclose,

- *wherein shedding a portion of tree buckets previously assigned to the new cache comprises: receiving load information from the new cache, the load information indicating whether the new cache system is overloaded; and using the load information to determine whether the new cache is overloaded.* (Li, abstract; col.1, lines 18-37, lines 38-54; col.2, lines 29-44, lines 48-61; col.6, lines 18-51; col.6, line 52 - col.7, line 53; col.9, lines 4-21; fig.6A; fig.9)

Response to Arguments

11. Applicant's arguments with respect to *claims 1, 18, 25, and 41* have been considered but they are not persuasive.

12. With regard to *claims 1, 18, 25, and 41*, the Applicants point out that:

- *In sum, Li fails to teach or suggest mechanisms for slowly assigning or shedding portions of either the full bucket allocation that was determined at startup, previously she buckets, or previously assigned buckets.*

However, the Examiner finds that the Applicants' arguments are not persuasive and maintains that Li and Dreszer disclose,

- *a) when a new cache system starts up in a cache cluster having a plurality of total buckets, determining a full bucket allocation for the new cache system;* (Li, abstract; col.1, lines 18-37, lines 38-54; col.2, lines 29-44, lines 48-61; col.6, lines 18-51; col.6, line 52 - col.7, line 53; col.9, lines 4-21; fig.6A; fig.9)

  Li teaches of a method for *"data redistribution process for adding a new node"* (Li, col.6, lines 18-19) to a database system *"where the new node is physically attached and registered to the parallel database network"* (Li, col.6, lines 20-21) and *"the buckets of data to be moved to the new node are determined for each existing node"* (Li, col.6, lines 24-25). Hence, when a new node is added to the system, a data redistribution process takes place, wherein a portion of the buckets of data for each existing node redistributes to the new node.

- *b) periodically determining a load of the new cache system;* (Li, col.1, lines 18-37-54; col.2, lines 29-44, lines 48-61; col.6, lines 18-51; col.6, line 52 - col.7, line 53; col.9, lines 4-21; fig.6A; fig.9)

  Li teaches that *"in the quiescent-mode operation, load balancing is the primary goal"* (Li, col.6, lines 52-53) implying that is very important to maintain a balanced workload among the existing nodes. Also, according to Li, *"while the description above has concentrated on a redistributing data when a new node is added into the parallel database system, the invention may also be used when the PDB system becomes imbalanced"* (Li, col.9, lines 4-7). Hence, Li not only teaches of redistributing the data buckets when a new node is added, but also using the technique once the system becomes imbalanced and thus requires a redistribution of the data buckets. Furthermore, Li admits as prior art *"eventually, the system will become imbalanced across the nodes. Thus, the data will*

occasionally have to be redistributed to rebalance the load" (Li, col.1, lines 50-52).

- *c) each time it is periodically determined that the new cache system is underloaded and buckets have not been previously shed from the new cache system,* (Li, col.1, lines 18-37-54; col.2, lines 29-44, lines 48-61; col.6, lines 18-51; col.6, line 52 - col.7, line 53; col.9, lines 4-21; fig.6A; fig.9)
  Li teaches that *"in the quiescent-mode operation, load balancing is the primary goal"* (Li, col.6, lines 52-53) implying that is very important to maintain a balanced workload among the existing nodes. Also, according to Li, *"while the description above has concentrated on a redistributing data when a new node is added into the parallel database system, the invention may also be used when the PDB system becomes imbalanced"* (Li, col.9, lines 4-7). Hence, Li not only teaches of redistributing the data buckets when a new node is added, but also using the technique once the system becomes imbalanced and thus requires a redistribution of the data buckets. Furthermore, Li admits as prior art that *"eventually, the system will become imbalanced across the nodes. Thus, the data will occasionally have to be redistributed to rebalance the load"* (Li, col.1, lines 50-52).
- *d) each time it is periodically determined that the new cache system is underloaded and buckets have been shed previously from the new cache system,* (Li, col.1, lines 18-37-54; col.2, lines 29-44, lines 48-61; col.6, lines 18-51; col.6, line 52 - col.7, line 53; col.9, lines 4-21; fig.6A; fig.9)
  Li teaches that *"in the quiescent-mode operation, load balancing is the primary goal"* (Li, col.6, lines 52-53) implying that is very important to maintain a balanced

workload among the existing nodes. Also, according to Li, *"while the description above has concentrated on a redistributing data when a new node is added into the parallel database system, the invention may also be used when the PDB system becomes imbalanced"* (Li, col.9, lines 4-7). Hence, Li not only teaches of redistributing the data buckets when a new node is added, but also using the technique once the system becomes imbalanced and thus requires a redistribution of the data buckets. Furthermore, Li admits as prior art that *"eventually, the system will become imbalanced across the nodes. Thus, the data will occasionally have to be redistributed to rebalance the load"* (Li, col.1, lines 50-52).

- *e) each time it is periodically determined that the new cache system is overloaded,* (Li, col.1, lines 18-37-54; col.2, lines 29-44, lines 48-61; col.6, lines 18-51; col.6, line 52 - col.7, line 53; col.9, lines 4-21; fig.6A; fig.9)
  Li teaches that *"in the quiescent-mode operation, load balancing is the primary goal"* (Li, col.6, lines 52-53) implying that is very important to maintain a balanced workload among the existing nodes. Also, according to Li, *"while the description above has concentrated on a redistributing data when a new node is added into the parallel database system, the invention may also be used when the PDB system becomes imbalanced"* (Li, col.9, lines 4-7). Hence, Li not only teaches of redistributing the data buckets when a new node is added, but also using the technique once the system becomes imbalanced and thus requires a redistribution of the data buckets. Furthermore, Li admits as prior art that *"eventually, the system will become imbalanced across the nodes. Thus, the*

data will occasionally have to be redistributed to rebalance the load" (Li, col.1, lines 50-52).

- *wherein each bucket portion corresponds to a portion of the total traffic being handled by the cache cluster.* (Li, abstract; col.1, lines 18-37, lines 38-54; col.2, lines 29-44, lines 48-61; col.6, lines 18-51; col.6, line 52 - col.7, line 53; col.9, lines 4-21; fig.6A; fig.9)

  Li teaches of a method for *"data redistribution process for adding a new node"* (Li, col.6, lines 18-19) to a database system *"where the new node is physically attached and registered to the parallel database network"* (Li, col.6, lines 20-21) and *"the buckets of data to be moved to the new node are determined for each existing node"* (Li, col.6, lines 24-25). Hence, when a new node is added to the system, a data redistribution process takes place, wherein a portion of the buckets of data for each existing node redistributes to the new node.

However, Li does not teach,

- *a) and assigning a portion of the full bucket allocation to the new cache system that was determined at start up;*
- *c) slowly assigning a portion of the full bucket allocation that was determined at startup unless the cache cluster is operating at a maximum load;*
- *d) slowly re-assigning a portion of the buckets that have been previously shed from the new cache system to the new cache system unless the cache cluster is operating at a maximum load; and*
- *e) shedding a portion of the buckets previously assigned to the new cache system,*

Dreszer teaches,

- *a) and assigning a portion of the full bucket allocation to the new cache system that was determined at start up;* (Dreszer, col.2, line 65 – col.3, line 16; col.3, lines 49-60; col.4, line 49 – col.5, line 8; col.5, lines 42-65; col.5, line 66 – col.6, line 8; col.6, lines 9-31; col.6, line 55 – col.7, line 15; col.8, lines 1-55; fig.10-12; col.7, lines 61-65; col.12, lines 49-67; modules 70-72, fig.4)

  Dreszer teaches a method of *"[allocating] of long term memory from the heap 34 during startup"* (Dreszer, col.8, lines 27-28) for a memory management system and, in particular, for a system cache buffer environment by *"adjusting the size in the found/existing allocation header 98 of the found block 95 (e.g., subtracting the requested size from the size of the allocation header 98) and creating a new allocation header 98a in memory beyond the portion 95a of the block 95 for the remaining free portion 95b of the found block 95"* (Dreszer, col.8, lines 42-47) and *"marking the existing allocation header 98 for said memory portion 95a as 'allocated' (not free)"* (Dreszer, col.8, lines 51-52). Hence, Dreszer teaches of allocating only portions of the available memory as necessary or when requested and adjusting the unallocated or the still available memory accordingly.

- *c) slowly assigning a portion of the full bucket allocation that was determined at startup unless the cache cluster is operating at a maximum load;* (Dreszer, col.2, line 65 – col.3, line 16; col.3, lines 49-60; col.4, line 49 – col.5, line 8; col.5, lines 42-65; col.5, line 66 – col.6, line 8; col.6, lines 9-31; col.6, line 55 – col.7, line 15; col.8, lines 1-55; fig.10-12; col.7, lines 61-65; col.12, lines 49-67; modules 70-72, fig.4)

  Dreszer teaches a method of *"[allocating] of long term memory from the heap 34 during startup"* (Dreszer, col.8, lines 27-28) for a memory management system

and, in particular, for a system cache buffer environment by *"adjusting the size in the found/existing allocation header 98 of the found block 95 (e.g., subtracting the requested size from the size of the allocation header 98) and creating a new allocation header 98a in memory beyond the portion 95a of the block 95 for the remaining free portion 95b of the found block 95"* (Dreszer, col.8, lines 42-47) and *"marking the existing allocation header 98 for said memory portion 95a as 'allocated' (not free)"* (Dreszer, col.8, lines 51-52). Hence, Dreszer teaches of allocating only portions of the available memory as necessary or when requested and adjusting the unallocated or the still available memory accordingly.

- *d) slowly re-assigning a portion of the buckets that have been previously shed from the new cache system to the new cache system unless the cache cluster is operating at a maximum load; and* (Dreszer, col.2, line 65 – col.3, line 16; col.3, lines 49-60; col.4, line 49 – col.5, line 8; col.5, lines 42-65; col.5, line 66 – col.6, line 8; col.6, lines 9-31; col.6, line 55 – col.7, line 15; col.8, lines 14-25; fig.10-12; col.7, lines 61-65; col.12, lines 49-67; modules 70-72, fig.4)

  Dreszer teaches of *"increasing/decreasing size queues in relation to memory requests (tuning) and performing trimming of size queues"* (Dreszer, col.7, lines 63-65) for a memory management system and, in particular, for a system cache buffer environment. Furthermore, Dreszer teaches of *"periodically reorganize/trim the size queue and attempt to release file system cache buffers back to the file system"* (Dreszer, col.12, lines 56-58).

- *e) shedding a portion of the buckets previously assigned to the new cache system,* (Dreszer, col.2, line 65 – col.3, line 16; col.3, lines 49-60; col.4, line 49 – col.5, line 8; col.5, lines 42-65; col.5, line 66 – col.6, line 8; col.6, lines 9-31;

col.6, line 55 – col.7, line 15; col.8, lines 14-25; fig.10-12; col.7, lines 61-65; col.12, lines 49-67; modules 70-72, fig.4)

Dreszer teaches of *"increasing/decreasing size queues in relation to memory requests (tuning) and performing trimming of size queues"* (Dreszer, col.7, lines 63-65) for a memory management system and, in particular, for a system cache buffer environment. Furthermore, Dreszer teaches of *"periodically reorganize/trim the size queue and attempt to release file system cache buffers back to the file system"* (Dreszer, col.12, lines 56-58).

Therefore, it would have been obvious to one of ordinary skill in the art at the time of the invention was made to the teachings of Dreszer with the teachings of Li to provide rapid memory allocation and de-allocation, reduced memory fragmentation, maximizes the amount of memory available for a cache (e.g., file system I/O buffers) while optimizing the amount of memory available for other uses, and manages competition for different memory uses by system self-adaptation to different usage levels across different network environments and over time within one network environment, including self-tuning to optimize performance to a variety of environments and dynamic conditions.

Therefore, the Applicants still failed to clearly disclose the novelty of the invention and identify specific limitation, which would define patentable distinction over prior art.

Conclusion

13. Any inquiry concerning this communication or earlier communications from the examiner should be directed to Thomas Duong whose telephone number is 571/272-3911. The

examiner can normally be reached on M-F 7:30AM - 4:00PM. If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Jason D. Cardone can be reached on 571/272-3933. The fax phone numbers for the organization where this application or proceeding is assigned are 571/273-8300 for regular communications and 571/273-8300 for After Final communications.

Thomas Duong (AU2145)

November 28, 2005

ZARNI MAUNG
SUPERVISORY PATENT EXAMINER